DE·i·GO

Migration Session

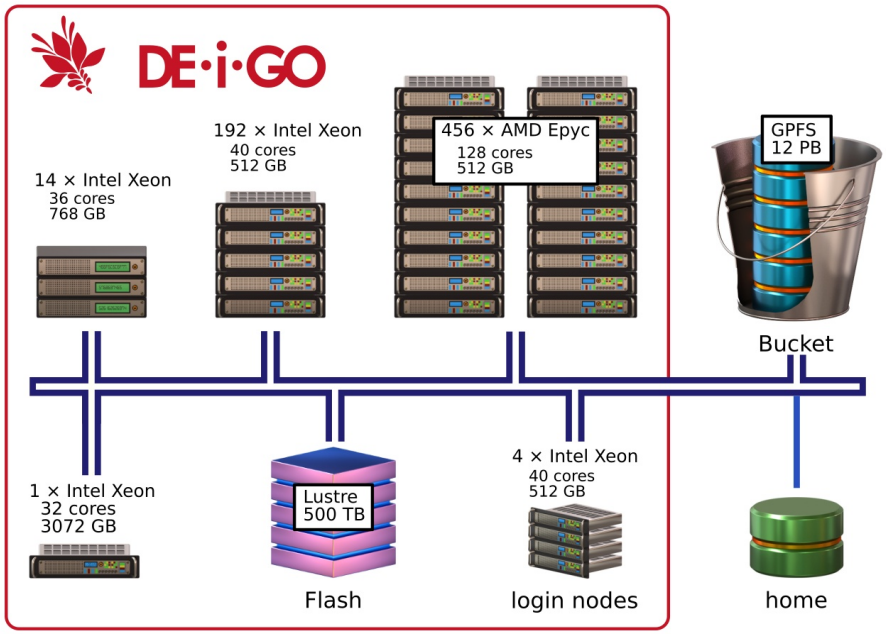
Jan Moren

Scientific Computing and
Data analysis section

OIST

Welcome to

DE·i·GO



The All-New OIST Computing Cluster

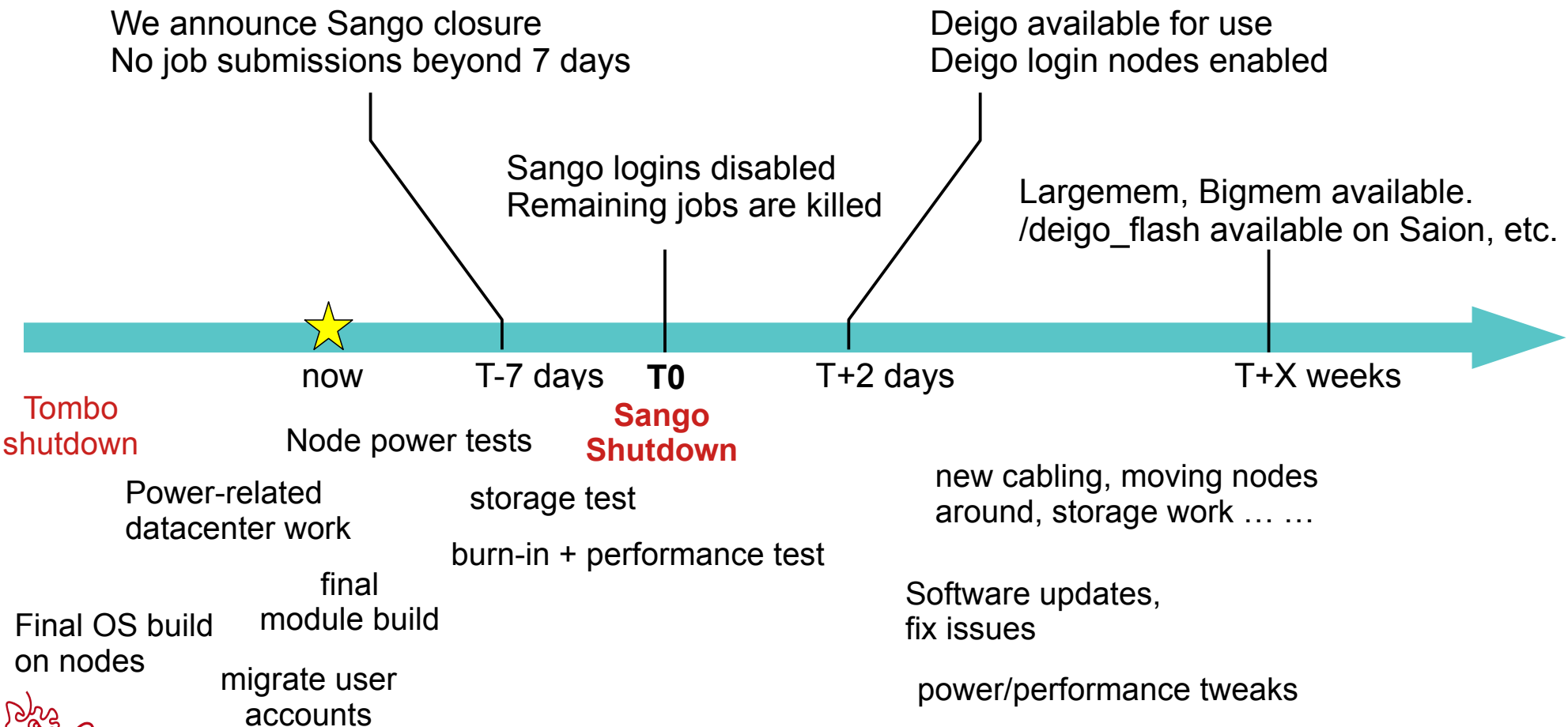
Welcome to

DE·i·GO

The BETA Release

The All-New OIST Computing Cluster

Timeline


We announce Sango closure
No job submissions beyond 7 days
Deigo available for use
Deigo login nodes enabled
Sango logins disabled
Remaining jobs are killed
Largemem, Bigmem available.
/deigo_flash available on Saion, etc.
now
T-7 days
T0
T+2 days
T+X weeks
Tombo shutdown
Sango Shutdown
Node power tests
Power-related datacenter work
storage test
burn-in + performance test
new cabling, moving nodes around, storage work … …
final module build
Final OS build on nodes
Software updates, fix issues
migrate user accounts
power/performance tweaks

Breaking News

We announce Sango closure
No job submissions beyond 7 days

Sango logins disabled
Remaining jobs are killed

Deigo available for use
Deigo login nodes enabled

Largemem, Bigmem available.
/deigo_flash available on Saion, etc.

Tombo shutdown

Friday July 3rd

Sango Shutdown

Friday July 10th

Monday July 13th

Today's Topics

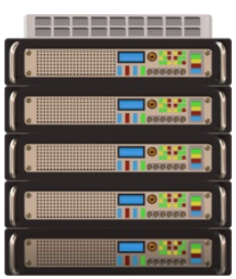
- Storage
  - What about Sango /work?
- Partitions
- Software modules
- Building Software on Deigo

Storage

Bucket

Sango /work

Compute nodes



Sango Workflow:

1) copy data to /work
2)
3)
4)

Storage

Bucket

Sango /work

Compute nodes

Sango Workflow:

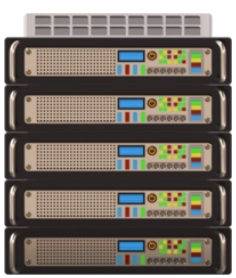
1) copy data to /work
2) Run job on the data
3)
4)

Storage

Bucket

Sango /work

Compute nodes



Sango Workflow:

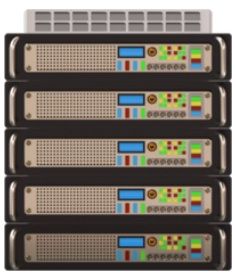
1) copy data to /work
2) Run job on the data
3) Copy results back
4)

Storage

Bucket

Sango /work

Compute nodes



Sango Workflow:

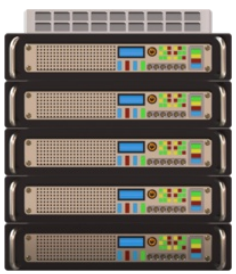
1) copy data to /work
2) Run job on the data
3) Copy results back
4) Clean up /work

Storage

Bucket

Sango /work

Compute nodes



Sango Workflow:

1) copy data to /work
2) Run job on the data
3) Copy results back
4) Clean up /work

But:

- Copying large data sets was cumbersome

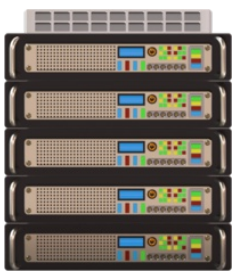
→ allocate space to leave current data on /work

Storage

Bucket

Sango /work

Compute nodes



Sango Workflow:

1) copy data to /work
2) Run job on the data
3) Copy results back
4) Clean up /work

But:

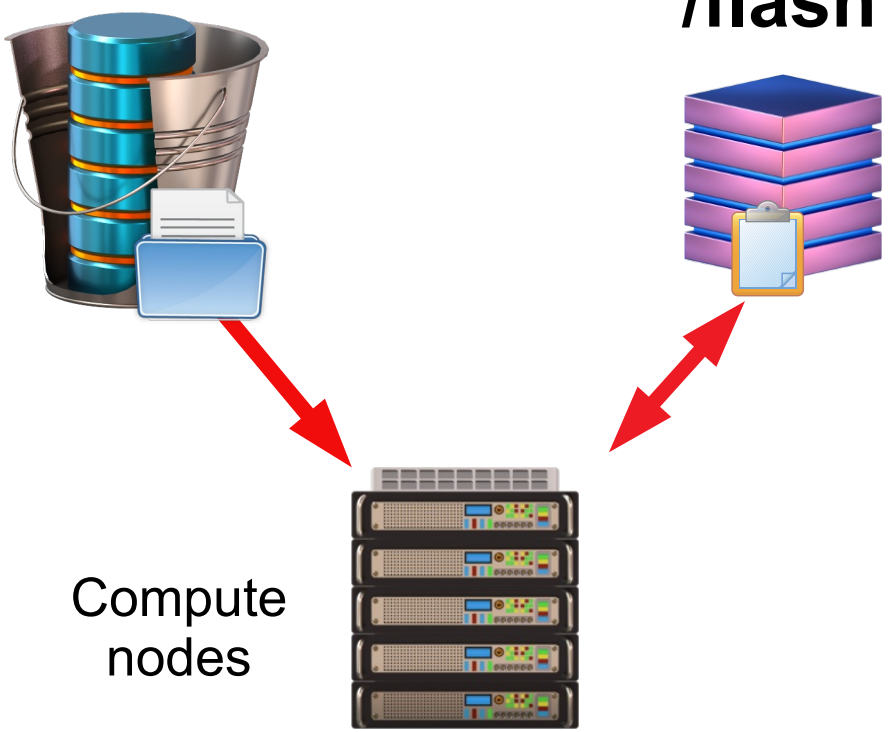
- Copying large data sets was cumbersome

→ /work became a dumping ground for old data

Storage

Bucket

Deigo /flash



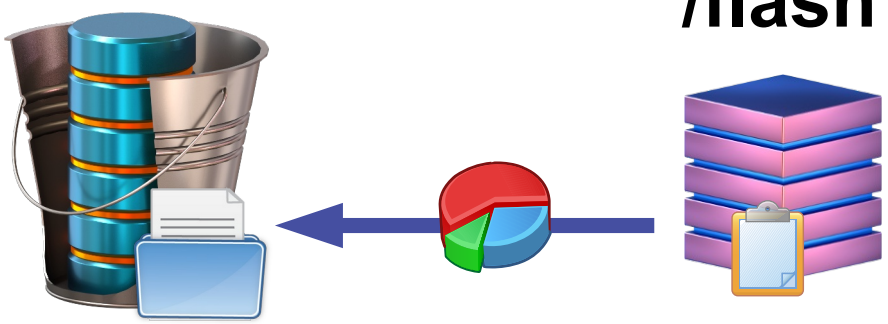
Deigo Workflow:

1) run job on data in Bucket
   use /flash for temporary storage
2)
3)

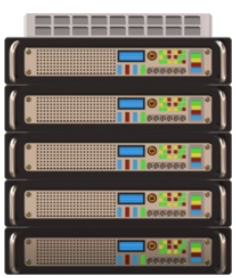
Storage

Bucket

Deigo /flash



Compute nodes



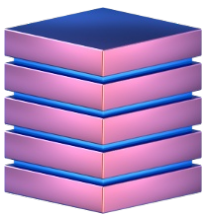
Deigo Workflow:

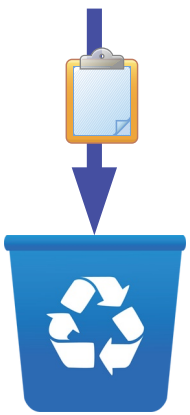
1) run job on data in Bucket
   use /flash for temporary storage
2) Copy results back
3)

Storage

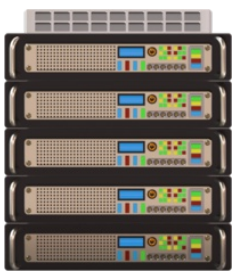
Bucket

Deigo /flash





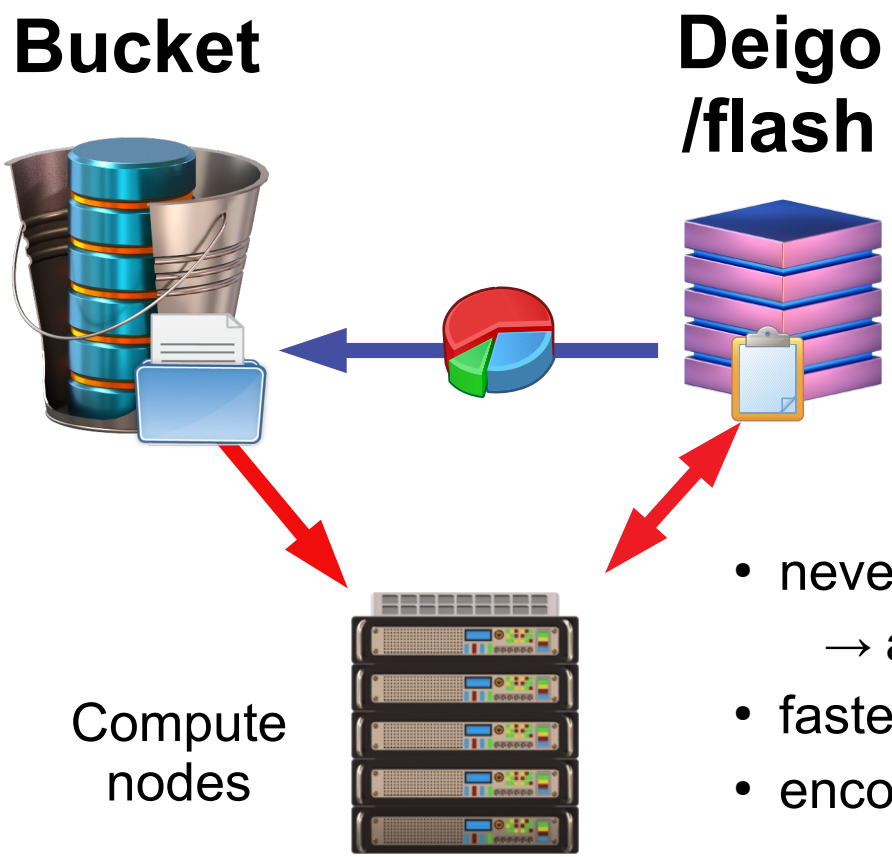
Compute nodes



Deigo Workflow:

1) run job on data in Bucket
   use /flash for temporary storage
2) Copy results back
3) Clean up /flash

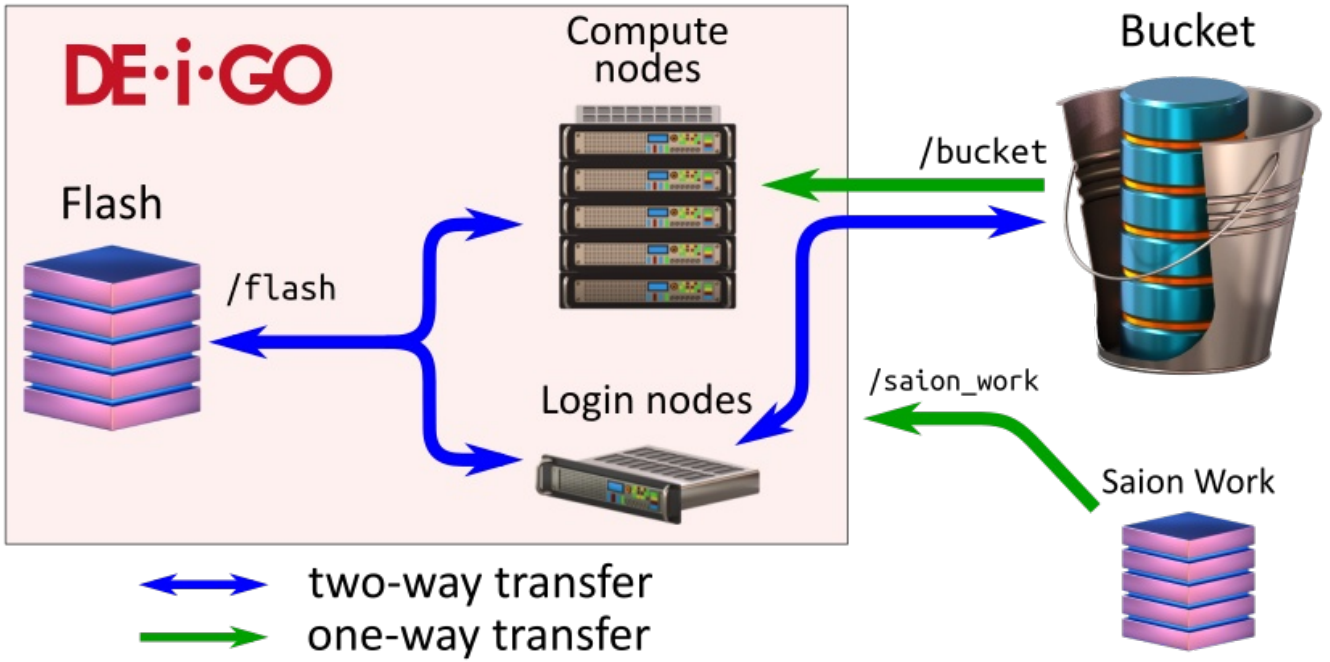
Storage



Deigo Workflow:

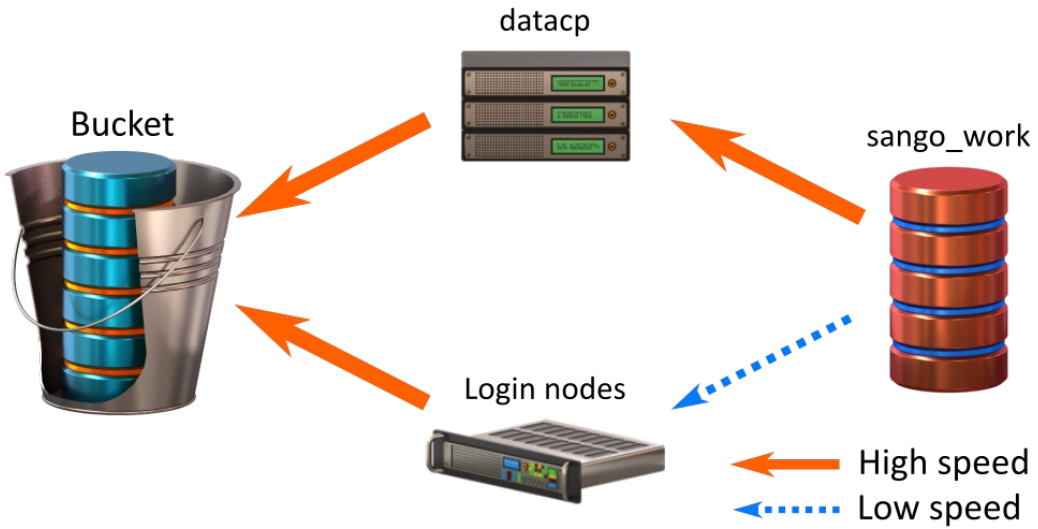
1) run job on data in Bucket
   use /flash for temporary storage
2) Copy results back
3) Clean up /flash

- never need to move data from bucket
  → always safe, in a single place
- faster storage, faster results
- encourages organisation



	Bucket		Deigo Flash		Saion Work	
capacity (per unit)	50TB+		10TB		10TB	
Deigo login nodes	/bucket	R/W	/flash	R/W	/saion_work	R
Deigo compute	/bucket	R	/flash	R/W	/saion_work	R
Saion login nodes	/bucket	R/W	/deigo_flash	R	/work	R/W
Saion compute	/bucket	R	/deigo_flash	R	/work	R/W

Sango Work



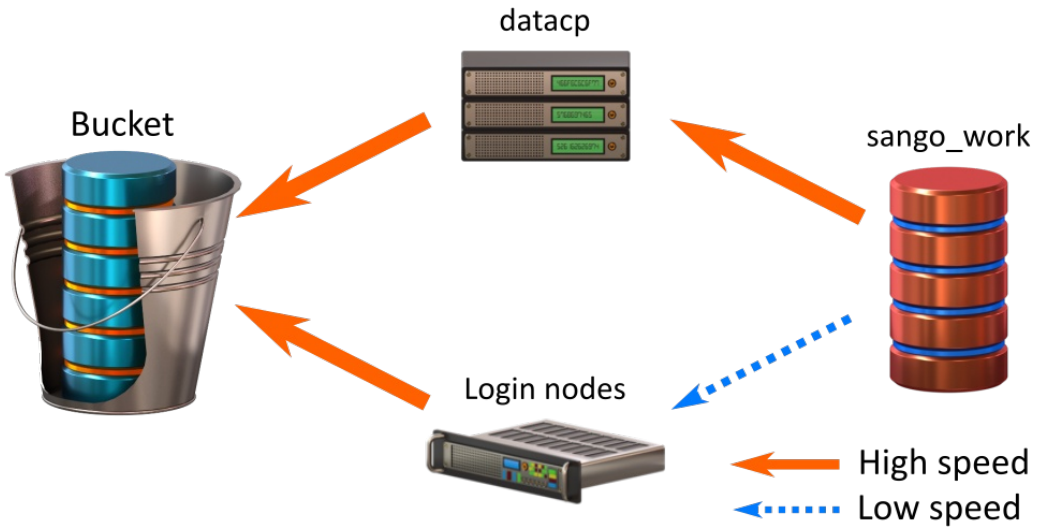
- read-write until Sango shutdown
- read-only as `/sango_work` on Deigo
  - We are preparing tools for copying and managing this data
- **Will Disappear** during 2020/2021

You must deal with this data

Focus on minimal disruption to daily work:

1. clean up Bucket
   — make space for moving data
2. move *currently used* data from /work
3. Deal with the rest

Sango Work

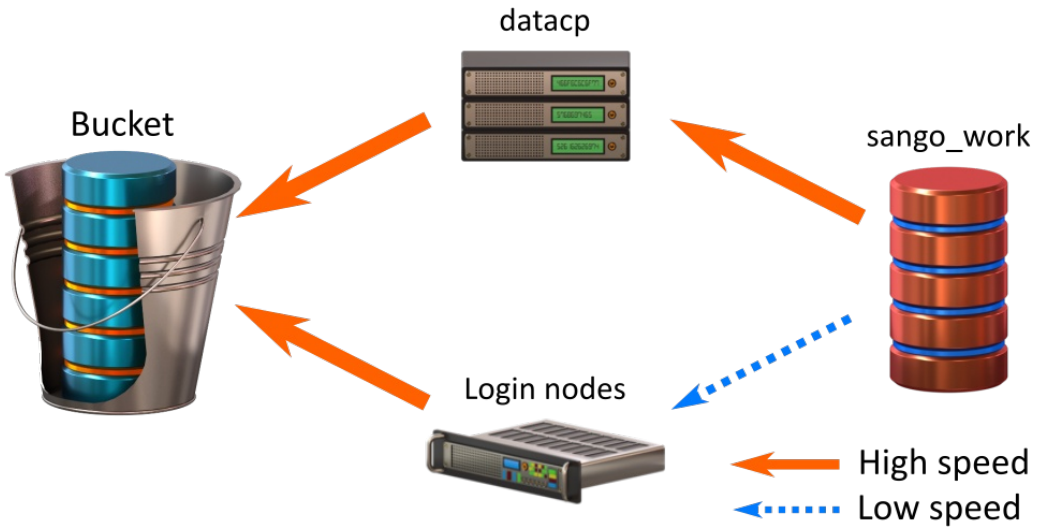


- read-write until Sango shutdown
- read-only as `/sango_work` on Deigo
  - We are preparing tools for copying and managing this data
- **Will Disappear** during 2020/2021

You must deal with this data

- If it's junk, delete it:
  - failed analyses or simulations
  - redundant copies of existing data
  - intermediate files, object files etc.
- Finished projects, former members:
  - Must be kept but will no longer be used
  - Ask us about archiving the data
- Currently used research data
  - Keep on Bucket

Sango Work



- read-write until Sango shutdown
- read-only as `/sango_work` on Deigo
  - We are preparing tools for copying and managing this data
- **Will Disappear** during 2020/2021

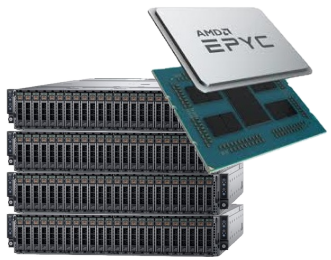
You must deal with this data

Directories with *lots and lots* of files

- Filesystems deal badly with lots of small files
- use “tar” to keep it as an archive
- *way* faster, more efficient, easier for you
- Copy large directories directly from sango Work into a tar archive on Bucket:

```
$ srun -p datacp -t 0-12 --pty bash
$ cd /sango_work/MyunitU
$ tar -czf /bucket/MyunitU/datadir.tgz datadir/
```

Deigo Hardware



456 AMD nodes

2 x EPYC 7702 2.0GHz
128 cores
512GB memory

58368 cores - 88%

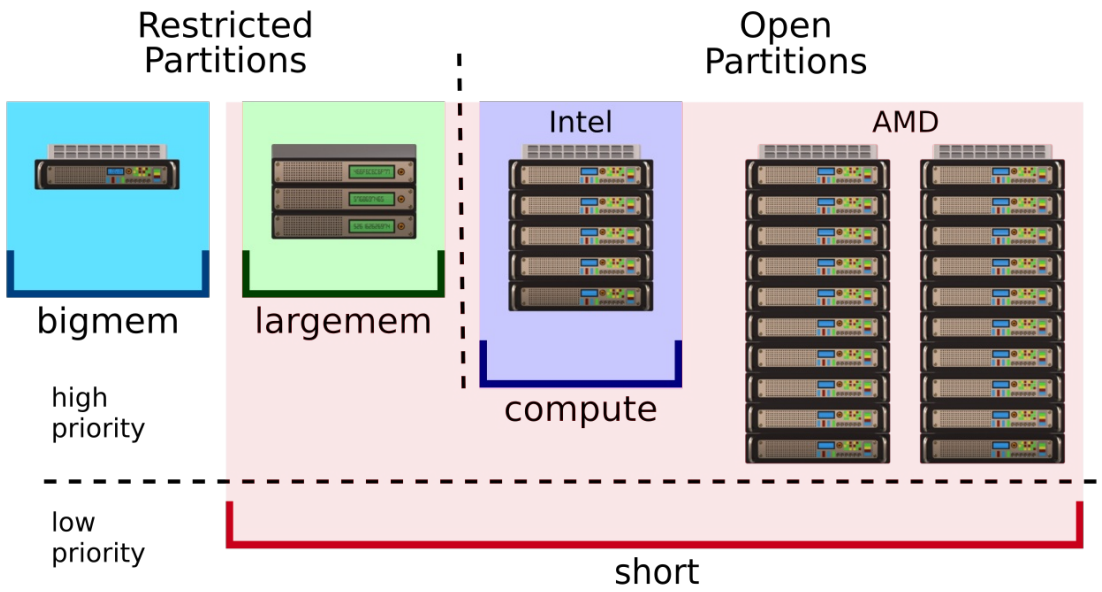
192 Intel nodes

2 x Xeon 6230 2.1GHz
40 cores
512GB memory

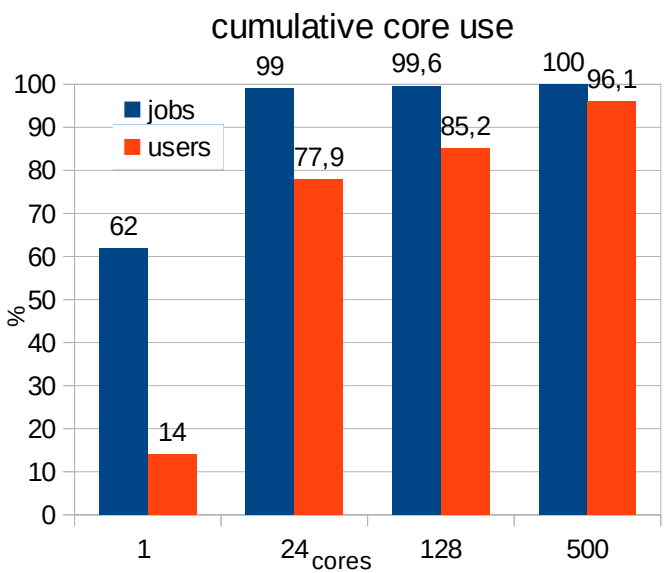
7680 cores - 12%

- Per *core*:
  - AMD is a bit faster for integer
  - AMD is faster for memory I/O
  - Intel is a bit faster FPU (esp. AVX512)
- Per *node*:
  - AMD trounces Intel
- example: matrix-matrix multiplication
  - Intel is 2× faster per core
  - AMD is faster per node (for large matrix)
- vector multiplication
  - AMD 2-5× faster per core (I/O bound)

Partitions



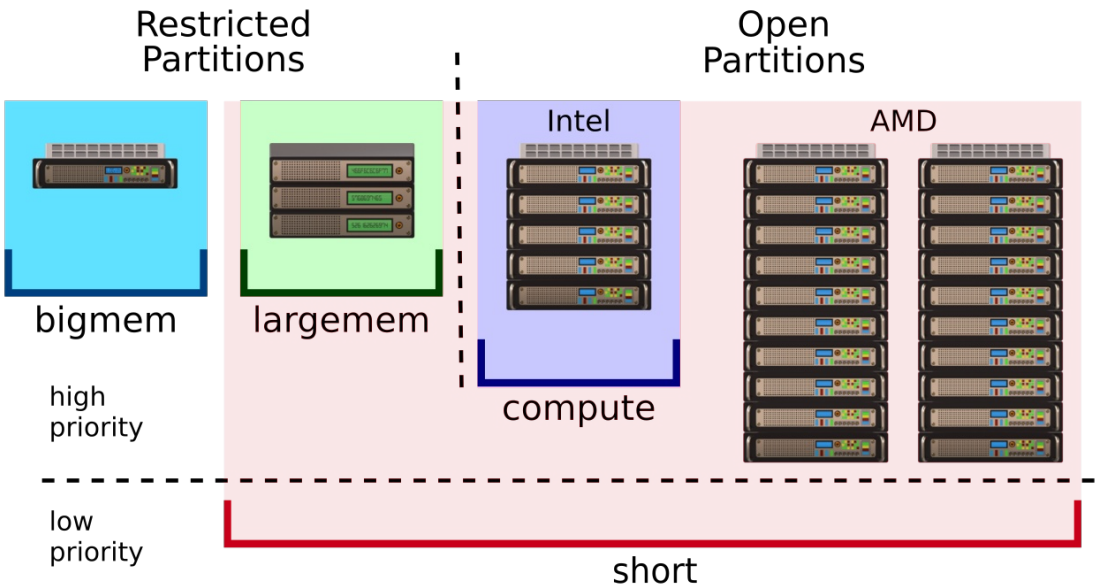
Cores	Jobs %	Users %
1	62	14
≤ 24	99	78
≤ 128	99.6	85
≤ 500	99.999	96



Partition	cores	time	memory	system
short	128	12 h	512G	All
compute	500	7 days	512G	Intel
largemem	~120	—	512/768G	generic
bigmem	8	—	3T	generic

OIST

Partitions



Partition	cores	time	memory	system
short	128	12 h	512G	All
compute	500	7 days	512G	Intel
largemem	~120	—	512/768G	generic
bigmem	8	—	3T	generic

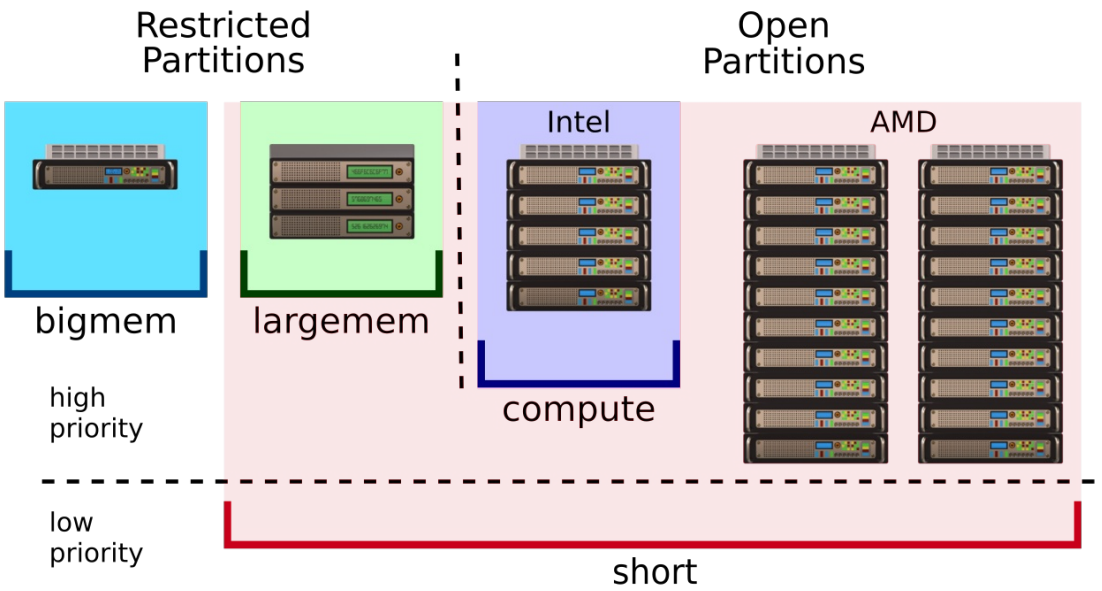
short

- all nodes - 66k cores
- low priority
  - if prio partition needs the node, job is suspended (will restart)
  - 4 hours *guaranteed* time
- For:
  - interactive jobs
  - array jobs
  - multithreaded jobs
  - **fits >90% of all our jobs**

compute

- intel nodes - 7.6k cores (~Sango)
- For:
  - long-running jobs
  - large jobs
  - optimised for Intel nodes

Partitions



Partition	cores	time	memory	system
short	128	12 h	512G	All
compute	500	7 days	512G	Intel
largemem	~120	—	512/768G	generic
bigmem	8	—	3T	generic

Running Jobs

- always specify the partition:

```
-p short
```

- Be careful with memory!
  - practical limit is 500G
  - default is 4G per core - if you ask for 128 cores you ask for 4*128=512G
- Don't waste cores
  - Find out - *test* - how many cores you can really use
- OpenMPI now uses PMIx as a process manager. Do:

```
srun --mpi=pmix <your program>
```

you can still use pmi2 if necessary:

```
srun --mpi=pmi2 <your program>
```

OIST

Software

Software modules are rebuilt

- Best effort - may be buggy.
- Only modules with “real use”
  - ~more than one user, more than a few instances over the past year
- Usually installed latest Sango version + latest released version
- Some software may not be identical to Sango, may not build at all on Deigo
  - ex: R, mpiblast

Software Order of Operations:

1. Try to use Deigo module
2. Try building new version (ask us)
3. Try rebuilding Sango version
4. Try Sango legacy modules
5. Try ‘sango’ container

Always: please ask us for help!

Software Modules

We now use Lmod modules

- Fully compatible with GNU Modules
- More convenenient for users
- Organise modules in groups
- Use Lua or TCL for module files
- many back-end improvements

https://lmod.readthedocs.io

‘ml’ command	‘module’ command	meaning
ml	module list	list loaded modules
ml av	module av	list available module files
ml julia	module load julia	load the module named ‘julia’
ml <command>	module <command>	all module commands can be used with ‘ml’

Software Modules

amd-modules	intel-modules	sango-legacy-modules	user-modules

Separate module areas:

- The default area
  most user software
- “amd-modules” and “intel-modules”
  built specifically for one CPU
  Intel compiler is in “intel-modules” but useful for all CPUs
- “sango-legacy-modules”
  the old Sango modules
  the “sango” container.
- “user-modules”
  user-managed modules

Software Modules

amd-modules	intel-modules	sango-legacy-modules	user-modules

The default area

most user software

“amd-modules” and “intel-modules”

built specifically for one type CPU

- “amd-modules” work everywhere
- “intel-modules” only on Deigo intel partition
  - *Intel compiler* is in “intel-modules” but useful for all CPUs
- largemem, bigmem are *not* “intel”

```
$ ml intel-modules
$ ml av
------------- /apps/.intel-modulefiles81 -----------------
   fftw.gcc/3.3.5                intel.mpi/2019_update5
   fftw.gcc/3.3.8         (D)    intel.mpi/2020_update1 (D)

. . .
------------------- /apps/.metamodules81 -----------------
   amd-modules           sango-legacy-modules
   intel-modules (L)     user-modules

-------------------- /apps/.modulefiles81 ----------------
   BUSCO/3.0.2                   java-jdk/11
   BUSCO/4.0.6           (D)     java-jdk/14         (D)

$ ml fftw.gcc/3.3.8
```

Software Modules

amd-modules	intel-modules	sango-legacy-modules	user-modules

Sango Legacy Modules

- The original, untouched Sango modules
- Some will work, many will not
  - complains about missing libraries
- **Unmaintained, use at your own risk**
- “sango” container can sometimes help
  - use “-m” to load modules
  - will currently not work with MPI apps

```
# load legacy modules, and the ‘sango’ container
$ ml sango-legacy-modules sango

# ‘pagan’ will not run on Deigo directly:
$ ml pagan
$ pagan
pagan: error while loading shared libraries:
libicudata.so.50: cannot open shared object file: No
such file or directory

# Use ‘sango’. use “-m <module>” to load modules:
$sango -m pagan pagan
PAGAN v.0.61 (23 July, 2015). (C) 2010-2014 by Ari
Löytynoja <ari.loytynoja@gmail.com>.
...
```

Software Modules

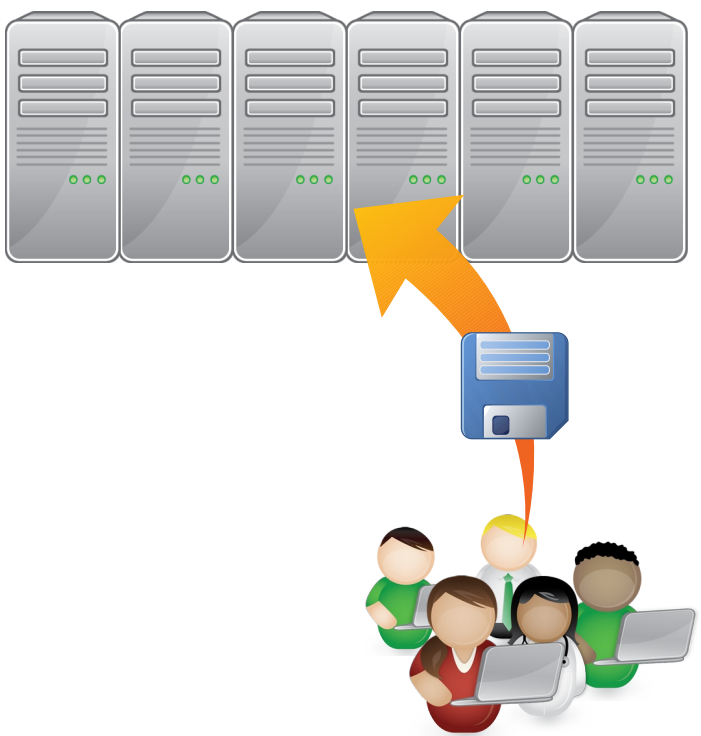
amd-modules	intel-modules	sango-legacy-modules	user-modules

user-modules

User-maintained modules for everybody

- For software in common use
- you know better than me how to install and maintain it
- **This is a real commitment** — we do *not* take over maintenance if you get bored

  → we strongly suggest you form a user group so you have multiple maintainers



Building Software

New “/apps/unit/” area set up

- reinstall or copy from “/sango_apps”
  - Always try to rebuild first! It’s faster, will work better
- /apps/unit now 50G limit per unit
  - plenty of space for software
  - do keep it cleaned up
  - Databases go in /bucket

Build software on a compute node

- Intel nodes have “AVX512” vector instructions
  - Software built on them may not work anywhere else — including largemem
  - login nodes are intel nodes!
  - Build on AMD nodes for most software
  - use intel nodes only for software that needs AVX512

```
# get part of an AMD node:
$ srun -t 0-4 -p short -C zen2 -c 16 --mem=8G --pty bash

# get part of an Intel node:
$ srun -t 0-4 -p short -C cascade -c 16 --mem=8G --pty bash
```

Building Software

We have three compilers:

gcc, Intel and AOCC

We have three BLAS libraries:

openBLAS, MKL and AOCL

If unsure, use latest GCC + OpenBLAS:

```
# Use latest gcc compiler:
$ ml gcc/9.1.1 OpenBLAS.gcc/0.3.9
```

painless installation, good overall performance

Intel Compilers

- Only 2019 and up will work on Deigo

OpenMPI

- Current version is 4.0.3
- 1.10 is ***ancient***.
  - can't use our high-speed network
  - doesn't scale
  - may fail
- MPI 3/4 have a few differences from 1
  1. build a newer version of the app
  2. patch the app to use MPI 4
     - It's quite easy! We can help.

Software Notes

Python 2 is dead.

- It is out of support and unmaintained
- Module only for legacy apps.
- **Do not use!**

Always use the major python version:

'python3', 'pip3', 'ipython3' etc

- '*python*' may refer to "'python 2' or 'python 3' on different systems.
- We have no plain 'python' on Deigo
  - software may need to be edited

Many Sango modules are unneeded

- newer versions of libraries and tools
- Available on Deigo by default

You can save default Lmod modules:

```
$ ml julia
$ ml save
# […  log out then in again …]
$ ml restore
# use multiple collections, short command form
$ ml gcc/9.1.1 OpenBLAS.gcc/0.3.9
$ ml s devel
...
$ ml r devel
```

Questions?